```
; Ampro Little Board BOOT ROM Assembly Source
; 30 mS Step Version
; 8080 Opcode Mnemonics - assembles with Digital Research ASM
;
;------- TMS July 2021 ------
```

```
;
; So be it :)
;
; ===== USING THE CODE =====
; The code uses standard 8080 opcode mnemonics. It can be assembled with the
; Digital Research ASM.COM that is standard with CP/M 2.2 and came on the Ampro
; Little Board system disk.
; It has been verified to produce a .HEX that produces a binary that is an
; identical match with code from the boot ROM dump.
;
; Note that assembling this source file creates an Intel .HEX file with base
; address of 08000H. Burning a new PROM or generating a .BIN from the .HEX
; requires mapping from the 8000H offset to 0000H offset in the PROM.
; This is typically accomplished by directing the programmer to map (load from)
; address 8000H in the .HEX file to offset 0000H in the PROM.
;
; ===== INTRO AND BACKGROUND =====
; This 8080 source code was created by the hand-disassembly of a
; ROM dump from an original Ampro Little Board boot ROM (p/n A75501-302)
; with labels, assembly directives, and functional annotations created
; by the author.
;
; When assembled the code reproduces the functional portion of the original
; ROM (offsets 00000H through 001C1H, inclusive). A full reproduction of the
; original ROM is made when the remaining unused bytes of the 4K ROM are set
; to 0FFh. As such, the output from this code may be considered incomplete
; with respect to generating a complete duplicate of the ROM.
;
; The hand-disassembly was actually first done using Z80
; opcode mnemonics. When the result appeared to only use 8080 functions,
; the Z80 version was run through a simple Z80 to 8080 opcode translator
; and assembly directives, labels, and comments added to create this file.
```

```
;
; This effort was made for educational purposes, help preserve a bit of
; history, and to serve as a possible aid to those looking to restore an
; original Little Board, or are simply curious about how this simple boot ROM
; functions.
;
; -- Binary Differences: "Maslin" vs this and other ROM sources
; This source code produces output that matches the ROM dump from an original
; Ampro Little Board p/n A75501-302 ROM as well as ROM binaries
; that have been uploaded by others for use with the MAME emulator software.
; This code and those uploads use a 30ms step rate for the WD1770 Floppy Disk
; Controller (FDC).
;
; The "Maslin" archives are another source for the boot ROM binary.
; The lb-bootv2 ROM code found in those archives does NOT match this code.
; That ROM uses a 6mS step rate for the FDC. This results in a a two-byte
; difference between the Maslin and A75501-302 binary as follows:
;
;       offset      maslin    mine    4k checksum
;   218 : 0x0da    0x08      0x0b    0xf01bb  ; FDC RESTORE command 6mS vs 30mS
;   376 : 0x178    0x58      0x5b    0xf01c1  ; FDC STEP IN command 6mS vs 30mS
;
; As of this writing it is unknown whether the Maslin version matched any
; production version of the Little Board ROM v2.0 and if so, whether it came
; before or after the version on my Little Board. The version number in the
; binary is the same between both. I believe my original Little Board to be a
; pretty late version since it was purchased in 1985 and  came with Revision F
; labeled on the system disk while the highest revision system disk for the
; original Little Board in the Maslin archives is Revision E.
; (There is a Rev. G, but Rev. G is for both the LB and LB+)
;
; -- STEP RATE - Does it matter?
; Either this version or the Maslin binary SHOULD work with MOST real
; (i.e. not emulated) Floppy Drives, but if there is any doubt check the
; specifications for the drive.
;
; The step rate only applies to operation of the boot ROM and does not lock
; down the step rate used once the system has booted. Ampro provides a
; CONFIG.COM utility that allows system software to be configured to use faster
; step rates during operation or even changed during run-time.
;
; -- Annotations
; The source code has been extensively commented. Additional annotations were
; made to aid the process of reverse assembly. Numbers in first column of the
; source comments list the physical offset in the original boot ROM. Numbers in
; parentheses list the corresponding address for the relocated code or RAM
; addresses.
;
; ===== BOOT ROM GENERAL OPERATION =====
; The basic operation of the boot ROM is as follows:
; 1. disables interrupts and clears the board control register
; 2. copies itself into RAM at 08000H and jumps to continue
;    execution using the RAM version
; 3. initializes the stack pointer to 09000H.
; 4. initializes the CTC channels 0 and 1 and DART channels A & B.
;    This sets up the CTC/DART combination for 9600 bps operation
;    on both DART channels
```

```
; 5. issues a "READ ADDRESS" command to the floppy controller to get
;    the track, side, sector number, sector length, CRC1, and CRC2
;    bytes into RAM
; 6. checks the sector number to determine whether it should
;    try to boot using sector number 1 or sector number 17. If
;    the sector number returned by READ ADDRESS is <17 then it
;    assumes sector 1 is the boot sector. Otherise it assumes
;    sector 17 is the boot sector
; 7. It then reads the boot sector into memory at 09000H and
;    jumps to it. Note that is overwrites the previous location of
;    of the stack, but at this point there is nothing to return to.
; 8. If the floppy controller reports a Record Not Found or CRC Error
;    at this point, or on any other command along the way in the boot
;    process, the boot code jumps to 0000H and does a complete reset.
;    If there's any problem with the drive or media,
;    the board enter an endless retry loop.
;
; ===== ADDITIONAL NOTES ON BOOT ROM OPERATION =====
; The Ampro Little Board boot code only uses programmed I/O for
; command and data transfer operations with the FDC. I.e. no DMA, no
; interrupts. This is generally OK with standard CP/M 2.2 since the
; operating system is intrinsically single threaded and blocks on storage
; transfer operations.
;
; The boot ROM includes a support routines, including a READ TRACK and STEP IN
; routine, that are not used by the boot ROM itself, but are used by
; the Ampro boot loader code on the disk. The standard Ampro boot loader code
; uses those routines to read the first two tracks off the floppy and into RAM
; on the Little Board at a RAM offset determined by the boot loader code.
;
; The READ TRACK support routine does not use the floppy controller's
; Read Track functionality. It reads in a "track" one sector at a time.
; If it finds that the sector length is 1024 bytes it assumes a track is five
; sectors, otherwise it assumes a track is ten sectors.
;
; ===== LIMITATIONS OF THE LITTLE BOARD BOOT ROM =====
; -- Lack of Error Status
; There's no console text or other output to confirm boot ROM execution or
; identify boot error status. It will just endlessly try to access the drive
; until it successfully reads track 0 sector #1 or #17 into memory,
; at which point it attempts to execute that sector without any integrity or
; validity checks. If there's a problem you will either see the floppy's LED
; 'blink' as it loops trying to read the sector and resets upon getting a
; CRC or Record Not Found error (which could be due to disk corruption,
; a blank disk, or an unexpected disk format) or the LED
; will go on and stay on - which typically indicates the processor is hung
; because it tried to execute a bad or incorrect boot sector, or failed
; later in the OS boot process.
;
; -- Boot Sector Number
; The boot ROM code was designed to boot ONLY from
; floppies where the boot sector is sector number 1 or 17 on Track 0.
; This is hard-coded.
;
; -- Support Routines
; The boot ROM offers only limited support routines and functionality
; that aid the on-disk boot loader reading the first two tracks off the
```

```
; disk. For example, there is STEP IN functionality, but no STEP OUT.
;
; -- MONITOR?
; This boot ROM code uses just 450 bytes out of the 4K EPROM. TBD whether
; there were ROM versions that did something useful with the space - such as
; offering a ROM monitor.
;
; -- 96TPI?
; Per the Ampro User's Manual:
; "Drive A must be a 48 tpi (40 track) 5-1/4-inch drive,
; either single- or double-sided."
;
; The User's Manual also states "system software is provided on a single-sided,
; 40-track format disk, which cannot be read by 96 tpi drives".
;
; It's not clear that DRIVE A must always be a 48 tpi drive. The boot code only
; boots from track 0 side 1 and the WD1770 Floppy Disk Controller has an 8-bit
; track register. The boot media must be MFM "double density" encoded
; at 250 kpbs.
;
; -- NO SINGLE DENSITY BOOT
; There is no support for booting from SINGLE DENSITY floppies.
; The DDEN# signal that controls whether the FDC is operating in single- or
; double-density mode is determined by a bit in the Little Board's Board
; Control register. That register is cleared at power-up and the boot ROM code
; never changes the DDEN# value. Software can set that bit to enable
; reading and writing single density disks AFTER booting.
;
; MISCELLANEOUS
; There appears to be a typo in the User's Manual. It says:
; "System Software reads each disk to determine the format (512 or 1024 bytes
; per sector), and whether single- or double-sided. Single sided disks have
; sector numbers from 1 to 10. Double-sided disk sector numbers are offset by
; 16 (16 to 25)". The text "(16 to 25)" should be "(17 to 26)".
;
;------------------------------------------------------------------------------
--
; I/O Address Equates:
;------------------------------------------------------------------------------
--
  BDCTLRG   EQU 000H ; Board Control Register
  FDCCMDRG  EQU 0C0H ; Floppy Disk Controller Command Register
  FDCSECRG  EQU 0C2H ; Floppy Disk Controller Sector Register
  FDCSTATRG EQU 0C4H ; Floppy Disk Controller Status Register
  FDCDATRG  EQU 0C7H ; Floppy Disk Controller Data Register
  CTCCH0RG  EQU 040H ; Counter/Timer Circuit Channel 0 Register
  CTCCH1RG  EQU 050H ; Counter/Timer Circuit Channel 1 Register
  DARTAREG  EQU 084H ; DART Channel A Register
  DARTBREG  EQU 08CH ; DART Channel B Register
;------------------------------------------------------------------------------
--
; Memory Address Equates
;------------------------------------------------------------------------------
--
  RAMLOC        EQU 08000H
  RESETVEC      EQU 00000H
  STACLOC       EQU 09000H
```

```
  BOOTSECDEST   EQU 09000H
;-------------------------------------------------------------------------
--
; Floppy Command Equates
;-------------------------------------------------------------------------
--
  FDCRESTORE    EQU 00BH  ; RESTORE, spin-up sequence disabled, 30 mS step rate
  FDCSTEPIN1    EQU 05BH  ; STEP IN side 1 with 30 mS step rate
  RDSGLSEC      EQU 088H  ; READ SINGLE SECTOR, no settling delay
  FDCRDADDR     EQU 0C8H  ; READ ADDRESS, no settling delay
  FDCTERMINATE  EQU 0D0H  ; FORCE INTERRUPT without interrupt
                          ; (D0H actually terminates FDC command in progress
                          ; without an interrupt)
;
;---------------------------------------
; JUMP TABLE
;---------------------------------------
  ORG RAMLOC
  JMP START-RAMLOC  ; 0000  Jump to beginning of boot code relative to 0 (029H)
  JMP SELDRIVE1     ; 0003  Reset FDC and select Drive 1 (080CEH)
  JMP STEPIN        ; 0006  Select Side 1 (if SIDE1FLAG non-0) and then issue
                    ;       STEP IN FDC command (08162H)
  JMP SELSIDE1      ; 0009  SELECT FLOPPY SIDE 1 in board control register.
                    ;       (081A3H)
  JMP TERMFCMD      ; 000C  TERMINATE FDC COMMAND in progress (08182H)
  JMP SETDESTADDR   ; 000F  SET DESTINATION ADDRESS (0819FH)
  JMP READTRACK     ; 0012  READ TRACK sector-by-sector (080E9H)
                    ;       READ 10 or 5 SECTORS starting from as selected by
                    ;       parameter 20 (sector length)
  JMP READSEC       ; 0015  READ SECTOR (08138H)
  JMP READADDR      ; 0018  ISSUE READ ADDRESS COMMAND (08121H)
;
;---------------------------------------
; PARAMETERS / VARIABLES
; The bytes in this area are used as variables and
; parameters after the ROM code has been copied to RAM.
; 0FFH is used as the power-on default value for the variables.
;---------------------------------------
PARMAREA:
;----------
SECDEST:  ; DEST ADDRESS FOR DATA FROM FDC READ SECTOR COMMAND
  DB 0FFH ; 001B  DESTINATION ADDRESS FOR SECTOR DATA LSB (08081BH)
  DB 0FFH ; 001C  DESTINATION ADDRESS FOR SECTOR DATA LSB (08081CH)
;----------
ADDRDEST: ; DEST ADDRESS FOR DATA FROM FDC READ ADDRESS COMMAND
          ; TRACK, SIDE #, SECTOR ADDR, SECTOR LENGTH, CRC1, CRC2
  DB 0FFH ; 001D DEST ADDRESS FOR READ ADDRESS DATA LSB (0801DH)   --- TRACK #
  DB 0FFH ; 001E DEST ADDRESS FOR READ ADDRESS DATA LSB (0801EH)    --- SIDE #
;----------
STARTSECFLAG:
  DB 0FFH ; 001F If > 17 boot sector #17 is used for boot sector # -- SECTOR #
          ;      if <= 17 then 0 is used for boot sector #     (0801FH)
;----------
SECLEN:   ;                                                    --- SECTOR
LENGTH
  DB 0FFH ; 0020 if equal to 003H (1024 bytes) SELECTS         (08020H)
          ;      "FIVE SECTOR" TABLE, otherwise "10 SECTOR" table is used.
```

```
;----------
  DB 0FFH ; 0021                                                (08021H) --- CRC1
  DB 0FFH ; 0022                                                (08022H) --- CRC2
;----------
BDCTLSHADOW:
  DB 0FFH ; 0023  BDCTLSHADOW (08023H)
;----------
PARM24:
  DB 0FFH ; 0024 Gets decremented once for every call to READFDCDATA (08024H)
;----------
TRKSECNUM:
  DB 0FFH ; 0025 Gets loaded with number for sectors as they're read by
          ;      READTRACK (08025H)
;----------
FDCSTAT:
  DB 0FFH ; 0026 Copy of the FDC STATUS value (08026H)
;----------
SIDE1FLAG:
  DB 0FFH ; 0027 non-0 = Select Side 1 before STEP IN (08027H)
;----------
SECOFFSET:
  DB 0FFH ; 0028  offset for sector numbers (08028h)
;----------------------------------------
;
;=========================================================
; START of boot code - execution starts from ROM at 00000H
; and then jumps to execution from memory after
; the ROM has been copied into RAM.
START:
  DI                  ; 0029  disable interrupts
  XRA A               ; 002A  clear Board Control Register
  OUT BDCTLRG         ; 002B
  ; Copy ROM to RAM at 08000H
  LXI H,RESETVEC      ; 002D  load source address
  LXI D,RAMLOC        ; 0030  load relocation destination address
RELOLOOP:             ; copy 4K bytes from source to destination
  MOV A,M             ; 0033
  STAX  D             ; 0034
  INX H               ; 0035
  INX D               ; 0036
  MOV A,H             ; 0037
  CPI 010H            ; 0038 4K written?
  JNZ RELOLOOP-RAMLOC ; 003A No. Move next byte
  ;
  JMP RAMCODE         ; 003D Yes. Jump over text
;----------------------------------------
CPYRT:            ; 0040 - 006D
  DB 'Boot Rom Version 2.0  (C) 1984 Ampro Computers'
;
;=========================================================
; The following code is executed from RAM
; after ROM has been copied to 08000H
;=========================================================
;
RAMCODE:
  LXI SP,STACLOC  ; 006E  init stack pointer
;
```

```
INITCTC:
  ; Init CTC channels 0 and 1 by writing sequence from CTCINITTAB
  ; (apply data from table until 0FFH encountered)
  LXI H,CTCINITAB ; 0071
INICTCLOOP:
  MOV A,M         ; 0074
  INR A           ; 0075
  JZ  INITDART    ; 0076
  DCR A           ; 0079
  OUT CTCCH0RG    ; 007A  Init ctr timer ch 0 = DART ch A baud rate gen
  OUT CTCCH1RG    ; 007C  Init ctr timer ch 1 = DART ch B baud rate gen
  INX H           ; 007E
  JMP INICTCLOOP  ; 007F
;
;----------------------------------------
CTCINITAB:        ; 082H (08082H)
  DB 047H         ; 047H => no interrupt, counter mode, 16 prescaler,
                  ;         falling edge, automatic trigger,
                  ;         time constant follows, software reset, control word
  DB 00DH         ; time constant 0D => 2MHz / $0D
                  ; yields 9615.38 (~9600 baud) when combined with /16
                  ; prescaler in DART
  DB 0FFH
;----------------------------------------
;
;=========================================================
; INITIALIZE DART
INITDART:
  LXI H,DARTINITAB  ; 0085
INIDARTLOOP:
  MOV A,M         ; 0088
  INR A           ; 0089
  JZ  BOOTSECX    ; 008A
  DCR A           ; 008D
  OUT DARTAREG    ; 008E
  OUT DARTBREG    ; 0090
  INX H           ; 0092
  JMP INIDARTLOOP ; 0093
;
;----------------------------------------
DARTINITAB:       ; 096h  (8096h)
  DB 04H
  DB 46H
  DB 05H,0EAH,03H,0C1H,0FFH
;----------------------------------------
;
;=========================================================
; GET AND EXECUTE BOOT SECTOR
; Read boot sector data into 09000H and execute it if successful
; otherwise, endless restart loop until successful
BOOTSECX:
  CALL SELDRIVE1    ; 009D  Select Drive 1 and return to Track 0
  CALL READADDR     ; 00A0
  CALL CHKFDCSTAT   ; 00A3
  LDA STARTSECFLAG  ; 00A6  Get parameter 1F (sector number from READADDR)
  CPI 011H          ; 00A9  Compare to 17 - If >= 17 boot sector #17 is used
                    ;       if < 17 then 0 is used.
```

```
  MVI A,000H        ; 00AB  Clear A (use move so Carry doesn't get changed)
  JC  NEXT3         ; 00AD  Leave A = 0
  INR A             ; 00B0  set A = 1
NEXT3:
  STA SIDE1FLAG     ; 00B1 Set SIDE1FLAG to 1
  ORA A             ; 00B4 Clear carry so it doesn't rotate in
  RAL               ; 00B5 If A = 1, A <= 16. If A = 0, no change.
  RAL               ; 00B6
  RAL               ; 00B7
  RAL               ; 00B8
  STA SECOFFSET     ; 00B9 Save Sector Offset (16 is default for boot ROM)
  LXI H,BOOTSECDEST ; 00BC
  SHLD SECDEST      ; 00BF
  INR A             ; 00C2 Starting sector number will be 17 or 1
  OUT FDCSECRG      ; 00C3 Write sector number to FDC Sector Register
  CALL READSEC      ; 00C5 Read the sector
  CALL CHKFDCSTAT   ; 00C8 Check status- restart if Record Not Found or CRC Err
  JMP BOOTSECDEST   ; 00CB  RUN BOOT SECTOR
;
;=======================================================
; SELECT AND RESET DRIVE 1 - Select Drive 1 and return
; to Track 0.
SELDRIVE1:
  MVI A,001H        ; 00CE  Assert Drive Select 1 and DDEN#.
                    ;       ROM enabled. All drives deselected.
  OUT BDCTLRG       ; 00D0
  STA BDCTLSHADOW   ; 00D2 Save to Shadow copy of Board Control Register
  CALL TERMFCMD     ; 00D5 Terminate any commands in progress
  MVI A,FDCRESTORE  ; 00D8 Tell FDC to return drive to Track 0.
  CALL ISSUFCMD     ; 00DA
WAITNOTBUSY:        ; wait until FDC command is finished
  IN  FDCSTATRG     ; 00DD
  RAR               ; 00DF
  JC  WAITNOTBUSY   ; 00E0 Check BUSY bit
  CALL LONGDELAY    ; 00E3 Delay required after issuing the RESTORE command
  JMP ODDRET        ; 00E6 Jump to a return instruction... Why?
;
;=======================================================
; READ TRACK - actually not a "READ TRACK" per se. Reads individual
; sectors from the present track in order listed in FIVESECTTAB
; or TENSECTTAB tables. The boot rom itself doesn't use this routine.
; It is used by Ampro boot loader code in the boot sector.
READTRACK:
  LDA SECLEN     ; 00E9 if equal to 003H SELECTS "FIVE SECTOR" TABLE,
                 ;      otherwise "10 SECTOR" table is used. (08020H)
  CPI 003H          ; 00EC
  LXI H,FIVESECTTAB ; 00EE
  JZ  RDSECLOOP     ; 00F1
  LXI H,TENSECTTAB  ; 00F4
RDSECLOOP:
  MOV C,M           ; 00F7 Read Sector Number from table into C
  INX H             ; 00F8 advance to next table entry
  INR C             ; 00F9 test for end of table (C = 0FFH)
  RZ                ; 00FA return if end of table
  DCR C             ; 00FB restore pre-test value of C
  LDA SECOFFSET     ; 00FC get starting sector offset
  ADD C             ; 00FF add sector number from table
```

```
  STA 08025H        ; 0100 save it
  OUT FDCSECRG      ; 0103  Write Sector Number to FDC
  PUSH  H           ; 0105
  CALL READSEC      ; 0106
  CALL CHKFDCSTAT   ; 0109 Check status- restart if Record Not Found or CRC Err
  POP H             ; 010C
  JMP RDSECLOOP     ; 010D
;
;---------------------------------------
TENSECTTAB:     ; (08110H)
    DB 01H,02H,03H,04H,05H,06H,07H
    DB 08H,09H,0AH,0FFH
;---------------------------------------
FIVESECTTAB:    ; (0811BH)
    DB 01H,02H,03H,04H,05H,0FFH
;---------------------------------------
;
;=======================================================
; READ FLOPPY ADDRESS field data and store the bytes at
; starting at ADDRDEST (Parameter "1D")
; This is used to load ANY sector number from the track
; into the STARTSECFLAG (parameter "1F") and to get
; the SECTOR LENGTH
READADDR:
  IN FDCSTATRG      ; 0121  clear floppy status register
  IN FDCDATRG       ; 0123  clear floppy data register
  CALL TERMFCMD     ; 0125  Terminate any FDC commands in progress
  LXI H,ADDRDEST    ; 0128  Load HL with ADDRDEST
  MVI A,FDCRDADDR   ; 012B  READ ADDRESS command
  CALL ISSUFCMD     ; 012D
  CALL RDFDCDATA    ; 0130
  MOV A,B           ; 0133
  STA FDCSTAT       ; 0134  Save FDC Status
  RET               ; 0137
;
;=======================================================
; READ SECTOR from floppy and store it where SECDEST is pointing
; Note that the FDC SECTOR REGISTER must be loaded with the
; sector number prior to calling this routine.
READSEC:
  IN FDCSTATRG      ; 0138  Read FDC Status
  IN FDCDATRG       ; 013A  Read FDC Data
  LHLD SECDEST      ; 013C  Get Destination Address into HL
  CALL TERMFCMD     ; 013F  Terminate any commands in progress
  MVI A,RDSGLSEC    ; 0142  FDC command to read a single sector
  CALL ISSUFCMD     ; 0144  Issue the command
  CALL RDFDCDATA    ; 0147  Read the data and store where SECDEST is pointing
                    ; until command completes because of no more data or error
  MOV A,B           ; 014A  Get FDC Status
  STA FDCSTAT       ; 014B  Save it in RAM
  SHLD SECDEST      ; 014E  Advance the Destination Address Value
  RET               ; 0151
;
;=======================================================
; READ DATA FROM FLOPPY UNTIL NOT BUSY
RDFDCDATA:
  IN  FDCSTATRG     ; 0152
```

```
  MOV B,A           ; 0154  Save FDC Status for examine after return
  RAR               ; 0155  FDC bit 0 BUSY BIT
  RNC               ; 0156  Return if NOT BUSY
  RAR               ; 0157  FDC bit 1 DRQ BIT, for reads 1b = data ready
  JNC RDFDCDATA     ; 0158  loop if data not ready
  IN  FDCDATRG      ; 015B  read the data
  MOV M,A           ; 015D  store at address given by HL
  INX H             ; 015E  increment HL
  JMP RDFDCDATA     ; 015F  loop until complete (not busy)
;
;=========================================================
; SELECT SIDE 1 if SIDE1FLAG non-0, then STEP IN to next track
STEPIN:
  LDA SIDE1FLAG     ; 0162  non-0 = SELECT SIDE 1 before STEP IN
  ORA A             ; 0165
  JZ  SASRTN        ; 0166
  CALL SELSIDE1     ; 0169 (081A3H)
  LXI H,PARM24      ; 016C  Decrement parameter 024H
  DCR M             ; 016F
  LXI H,000A7H      ; 0170  Set interval for delay
  JMP DELAY         ; 0173  delay and RETURN
SASRTN:             ; STEP IN, get status, and RETURN
  MVI A,FDCSTEPIN1  ; 0176  STEP IN, SIDE 1, update track reg, 30 mS step rate
  CALL ISSUFCMD     ; 0178
INPROGRESSWAIT:     ; wait for BUSY to be asserted before returning
  IN  FDCSTATRG     ; 017B
  RAR               ; 017D
  JC  INPROGRESSWAIT  ; 017E
  RET               ; 0181
;
;=========================================================
; TERMINATE FDC COMMAND IN PROGRESS
TERMFCMD:
  MVI A,FDCTERMINATE  ; 0182
  OUT FDCCMDRG      ; 0184
  MVI A,000H        ; 0186
DELAY255:
  DCR A             ; 0188
  JNZ DELAY255      ; 0189
  IN  FDCSTATRG     ; 018C
  RET               ; 018E
;
;=========================================================
; ISSUE COMMAND TO FDC and wait until it's in process
ISSUFCMD:
  OUT FDCCMDRG      ; 018F
  MVI A,00AH        ; 0191 very short delay (in uS) before reading status
TINYDELAY:
  DCR A             ; 0193
  JNZ TINYDELAY     ; 0194
FBUSYWAIT:
  IN  FDCSTATRG     ; 0197
  ANI 001H          ; 0199
  JZ  FBUSYWAIT     ; 019B
  RET               ; 019E
;
;=========================================================
```

```
; SET FDC DATA DESTINATION ADDRESS in SECDEST variable
SETDESTADDR:
  SHLD SECDEST      ; 019F  Set parameter 1B with contents of HL
  RET               ; 01A2
;
;=========================================================
; SELECT FLOPPY SIDE 1 in board control register. Leave other bits unaltered.
SELSIDE1:
  LDA BDCTLSHADOW   ; 01A3
  ORI 010H          ; 01A6  FLOPPY SIDE 1 bit
  OUT BDCTLRG       ; 01A8
  RET               ; 01AA
;
;=========================================================
; SELDRIVE1 RETURN - The SELDRIVE1 routine returns by jumping here.
; Unknown whether it's used for anything else.
ODDRET:
  RET               ; 01AB
;
;=========================================================
; CHECK FDC STATUS - Reset if Record Not Found or CRC error. Otherwise, return.
CHKFDCSTAT:
  LDA FDCSTAT       ; 01AC  Get last FDC status
  ANI 018H          ; 01AF  Test for Record Not Found or CRC error
  RZ                ; 01B1  return if no errors
  XRA A             ; 01B2  otherwise clear the board control register
  OUT BDCTLRG       ; 01B3
  JMP RESETVEC      ; 01B5 and restart the ROM. Note that restarting
                    ;      from 0 also restores any RAM variable/parameters
                    ;      to their power-on defaults
;
;=========================================================
; A DELAY with a rather specific loop count of 0413CH.
LONGDELAY:
  LXI H,0413CH      ; 01B8  Load constant for fixed long delay loop
; DELAY is also an entry point using HL as an input parameter
DELAY:
  DCX H             ; 01BB
  MOV A,H           ; 01BC
  ORA L             ; 01BD
  JNZ DELAY         ; 01BE
  RET               ; 01C1
```